Utilization of Path-Clustering in Efficient Stress-Control Gate Replacement for NBTI Mitigation

AUTHOR(S):

MORITA, Shumpei; BIAN, Song; SHINTANI, Michihiro; HIROMOTO, Masayuki; SATO, Takashi

CITATION:

MORITA, Shumpei ...[et al]. Utilization of Path-Clustering in Efficient Stress-Control Gate Replacement for NBTI Mitigation. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences 2017, E100.A(7): 1464-1472

ISSUE DATE:

2017-07-01

URL:

http://hdl.handle.net/2433/226311

IEICE TRANS. FUNDAMENTALS, VOL.E100–A, NO.7 JULY 2017

PAPER *Special Section on Design Methodologies for System on a Chip*

Utilization of Path-Clustering in Efficient Stress-Control Gate Replacement for NBTI Mitigation

Shumpei MORITA[†a)], ***Student Member***, **Song BIAN**[†], ***Nonmember***, **Michihiro SHINTANI**[†], **Masayuki HIROMOTO**[†], ***and*** **Takashi SATO**[†], ***Members***

SUMMARY Replacement of highly stressed logic gates with internal node control (INC) logics is known to be an effective way to alleviate timing degradation due to NBTI. We propose a path clustering approach to accelerate finding effective replacement gates. Upon the observation that there exist paths that always become timing critical after aging, critical path candidates are clustered to select representative path in each cluster. With efficient data structure to further reduce timing calculation, INC logic optimization has first became tractable in practical time. Through the experiments using a processor, 171x speedup has been demonstrated while retaining almost the same level of mitigation gain.

key words: *NBTI mitigation, reliability, transistor aging, performance degradation, internal node control*

1. Introduction

The continuous shrinkage in transistor sizes posits reliability problems as one of the most important topics in semiconductor industry [1]. In particular, negative bias temperature instability (NBTI) is one of the major threats to the reliability. NBTI is a phenomenon that increases the threshold voltage (V_{th}) of pMOS transistors by applying a negative bias to the gate electrode. The condition to give rise to increase V_{th} is called the stress phase for NBTI degradation. Meanwhile, the V_{th} degradation recovers toward its original value when the gate electrode is at zero bias, known as the recovery phase. The recovery of the V_{th} is not necessarily complete, hence the repetition of the stress and recovery phases eventually imposes the device a long-term delay degradation.

There are intensive studies in the field related to NBTI degradation [2]–[6]. The NBTI degradation is related to the average frequency of stress phase (signal probability). As shown later, a signal probability close to 1.0 significantly increases threshold voltage. Even a slight reduction of the signal probability will provide a substantial remedy for the transistors to suppress threshold voltage increase. For that reason, mitigation techniques by controlling signal probability have been proposed.

In [4], [5], the internal node control (INC) technique is described. This technique replaces an existing logic gate that is upstream on the stressed gate with the INC logic, which is functionally equivalent to the original one but adding a mitigation signal input to force its output value to be logical zero. With the mitigation signal, INC logic decreases the signal probabilities of downstream gates. Using this technique, NBTI mitigation can be achieved. In order to avoid losing logic states, these techniques need to be applied when the circuit is in a sleep mode. In contrast with these, a technique that practices NBTI mitigation when no operation (NOP) instruction is brought out is proposed in [6]. This enables the mitigation of NBTI even when the processor is working.

However, NBTI mitigation techniques by the INC logic replacement require optimization to find effective gates that minimize the power and area overheads while obtaining the mitigation gain. Finding effective gates to replace with the NBTI mitigation logic involves evaluations of signal probability and aged delay. These calculations are highly CPU intensive, but have to be conducted repeatedly during the optimization process because the replacement of a gate will drastically alter the signal probabilities of the surrounding logic gates. In addition, modern circuits can be extremely large, making it difficult to find the optimal replacement gates in a practical time.

In this paper, we propose an efficient method to execute effective gate replacements with INC logic. In the proposed method, critical path candidates are clustered on the basis of the similarity in gate instances, to find the *representative path* in each cluster. In order to quantify the similarity between paths, the signature of the path is defined by the gate-instance vector. Two-stage clustering approach, by using Newman algorithm [7] and agglomerative hierarchical clustering [8], has been proposed to balance efficiency and clustering accuracy.

With the idea of the representative path along which the gate replacements are conducted, the number of candidate gates for INC replacement is reduced. Time required for delay evaluation is thus significantly shortened as the gates on the representative paths or the critical paths are only considered during the optimization loop. The proposed clustering-based approach still captures the logic gates that constitute the most critical paths after aging. Also, the changes in critical paths due to aging and due to the INC logic replacements are correctly tracked with the updates of the representative paths. Hence, mitigation efficiency is preserved. Also, during optimization, *gate pruning* is conducted to further reduce the necessity of delay calculation. The path that is affected by the replacement of a gate is recorded beforehand so that

Manuscript received September 14, 2016.
Manuscript revised January 24, 2017.

[†]The authors are with Department of Communications and Computer Engineering, School of Informatics, Kyoto University, Kyoto-shi, 606-8501 Japan.
a) E-mail: paper@easter.kuee.kyoto-u.ac.jp

DOI: 10.1587/transfun.E100.A.1464

redundant delay calculations are skipped.

Through numerical experiments, it is shown that our approach significantly reduces the computational time for the INC logic replacement. As compared to the naive approach that handles all possible replacement candidates (COMP in Sect. 3), the proposed method reduced the time required for delay evaluation to 1.84% (from 8,877 to 163.5 minutes), and the total time required for entire optimization procedure has been accelerated by 171x compared to the conventional method.

This paper is an extended work from our previous work [9]. We have newly introduced the aforementioned two-stage clustering and the gate pruning techniques, which contribute to further acceleration of the calculation time to replace INC logics. The key contribution of this work is summarized as follows:

- An efficient and accurate method for INC logic replacement for NBTI mitigation has been presented. Acceleration of 171x has been achieved while maintaining the accuracy, making the INC logic replacement applicable to practical circuits.
- Two-stage hierarchical path clustering algorithm and the concept of gate pruning have been proposed, which are suitable for extracting the representative paths along which most important gates for INC-based NBTI mitigation.
- Combination of the above techniques first made it possible to update signal probability and path delay during INC replacement optimization.

This paper is organized as follows. In Sect. 2, the NBTI mitigation using INC logic and its procedural flow is described. The proposed replacement methods of INC logic are introduced in Sect. 3. Path clustering approach is explained in more detail in Sect. 4. The numerical experiment using a RISC processor is shown in Sect. 5 to quantitatively evaluate the effectiveness of the proposed method. This paper concludes in Sect. 6.

2. NBTI Mitigation

2.1 NBTI Model and Mitigation

The reaction-diffusion (RD) model is widely accepted to predict NBTI-induced threshold voltage degradation [2]. The RD model is based on the reaction-diffusion mechanism caused by hydrogen ions separated from Si-SiO_2 interface. It well reproduces the long-term NBTI-induced degradation [1]. The ions diffuse into the gate oxide film when the negative bias is applied to the gate electrode. The hydrogen ion recombines with the interface of the silicon when the application of the negative bias is removed. This recombination causes threshold voltage recovery. For that reason, the signal probability is an important factor to evaluate NBTI degradation in the logic circuits. In [2], [3], when the clock period T_{clk} is short enough, threshold voltage degradation due to NBTI is modeled as follows:

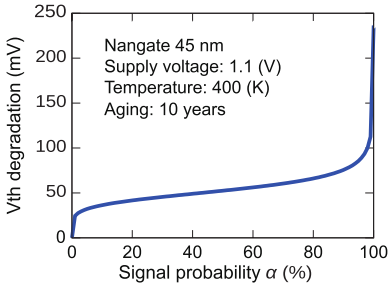

Fig. 1 Relationship between the signal probability and the threshold voltage degradation of the Nangate Open Cell Library [10].

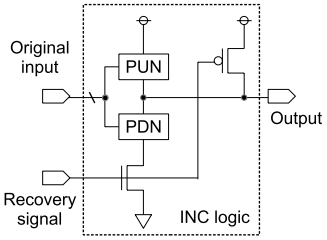

Fig. 2 General structure of the INC logic.

$$|\Delta V_{\mathrm{th}}(t)| \approx \left(\frac{0.001n^2K_v^2\alpha Ct}{0.81t_{\mathrm{ox}}^2(1-\alpha)}\right)^n, \tag{1}$$

where K_v is a function of gate-source voltage V_{gs} and threshold voltage V_{th}. α and t are the signal probability and time, respectively. C is a function of temperature. Typical value of n is 1/6. Equation (1) can be only applied to AC signals. Since Eq. (1) diverges to infinity when α approaches 1, we use an upper limit of the NBTI degradation for DC signals as shown in [3]. The upper limit is defined as follows:

$$|\Delta V_{\mathrm{th}}(t)| = (K_v^2 t)^n. \tag{2}$$

Applying Eqs. (1) and (2) to the Nangate 45 nm Open Cell Library [10], the relationship between the signal probability and the threshold voltage degradation is obtained as shown in Fig. 1. V_{th} seriously degrades when the stress probability is close to 1.0. In that region, the threshold voltage degradation can be greatly reduced when the stress probability is lowered even by a small amount. In the following, the gate whose output signal probability is greater than 0.9 is referred to as stress gate. As will be later explained in Sect. 5.1, the probability less than 0.1 should also be considered as stress gate. However, due to probability propagation along the paths, considering the case greater than 0.9 can yield sufficient mitigation results.

2.2 Mitigation of Circuit Degradation by INC Logic

In [4]–[6], the stress probability is reduced by forcing the gate input to be logical "1." In these papers, the selected logic gates in a circuit are replaced with INC logics so that the input signal probability of the downstream logic gates is controlled. Figure 2 shows the general structure of INC logic. A pMOS transistor and an nMOS transistor are added

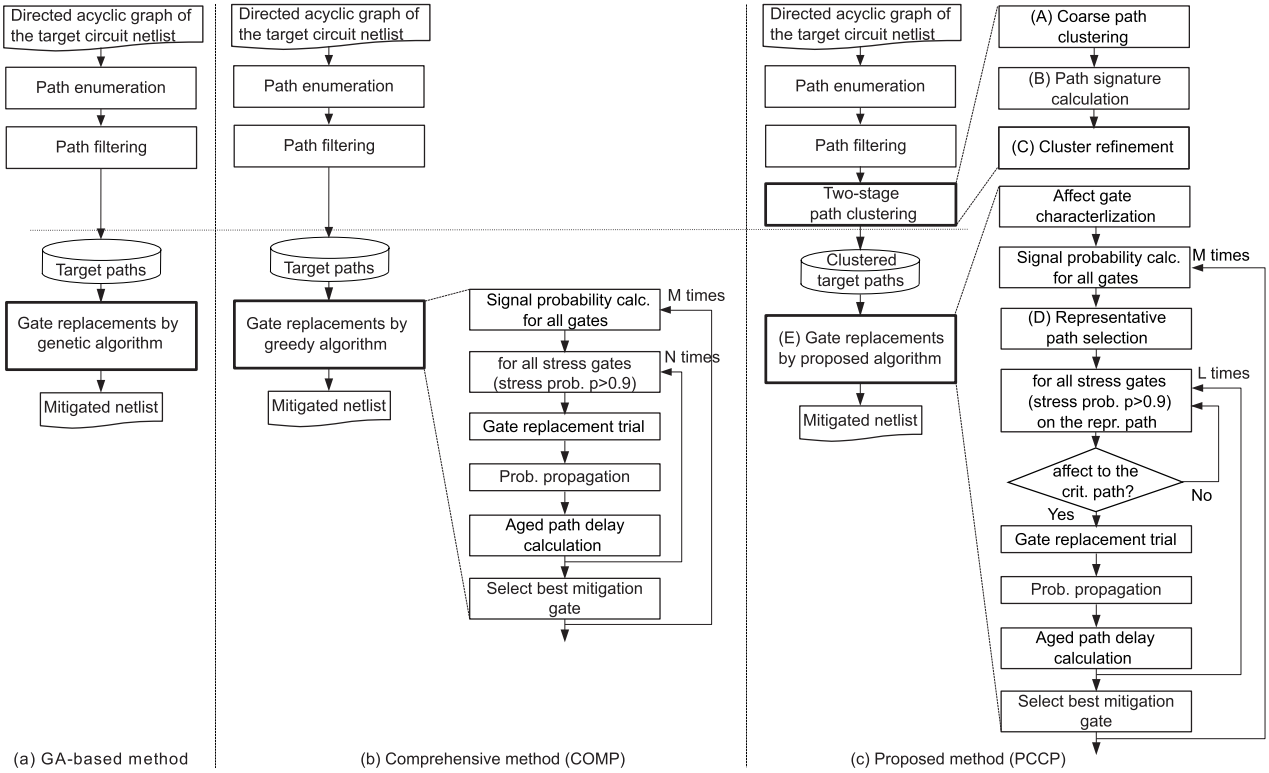

Fig. 3 Three methods of INC logic replacement.

to the original logic gate. The additional pMOS transistor connects output node to the supply rail in parallel to the pull up network (PUN) of the original logic gate. Also, an nMOS transistor is added in series to the pull down network (PDN) to cut off the connection to the ground network. The added transistors are controlled by the recovery signal. When the recovery signal is de-asserted (logical "1"), the INC logic functions completely equal to the original logic gate. When the recovery signal is asserted (logical "0"), the output of the INC logic is forced to be "1" regardless of the logic values of the original input signals. By asserting the recovery signal, the signal probability of the direct downstream gates decreases and the degradation is mitigated. In [6], stress gates are estimated in advance, and those gates are replaced to INC logics.

3. INC Logic Replacement

In [6], NBTI mitigation through the INC logic replacement is proposed. The conventional optimization flow for the INC logic replacement is shown in Fig. 3(a). Here, the optimization objective is to find the replacement gates, which are stress gates and replacement of which most alleviates the worst delay of the aged circuit.

In the conventional flow shown in Fig. 3(a), a signal flow of the gate-level netlist of the test circuit is first represented by a directed acyclic graph (DAG). Then, paths in the circuit are enumerated. On the basis of timing criticality, the paths are filtered to reduce the number of paths considered to apply INC replacement. Then, using genetic algorithm (GA), M logic gates are simultaneously replaced with respective INC logics. During the optimization, the initial signal probabilities have been used. Note that the signal probability calculation is performed at the transistor level. Here, as was noted in [6], replacing a gate with INC logic may cause significant change in the signal probability of a large number of gates in the circuit, which is referred to as *butterfly effect*. This chaotic behavior makes it significantly difficult to correctly estimate the efficacy of the replacement if signal probabilities are not recalculated during optimization because path delay greatly changes according to the change of the signal probability. However, considering the optimization time to finish within a practical limit, the recalculations of probability and aged path delay are omitted in the conventional work. This may have degraded the quality of optimization result substantially.

We propose to update the signal probability and to calculate the aged path delay during the optimization process to improve the quality of optimization result. Figure 3(b) shows a simple comprehensive implementation that includes the probability update and the worst path delay calculation. In Fig. 3(b), an optimal gate replacement candidate is selected among all logic gates in the critical path candidates in an exhaustive manner, i.e., by evaluating the efficacy of each replacement. It is widely known that the comprehensive algorithm does not give a globally optimal result in most cases. However, as shown later in the experiment section, the comprehensive algorithm in Fig. 3(b) yields much better

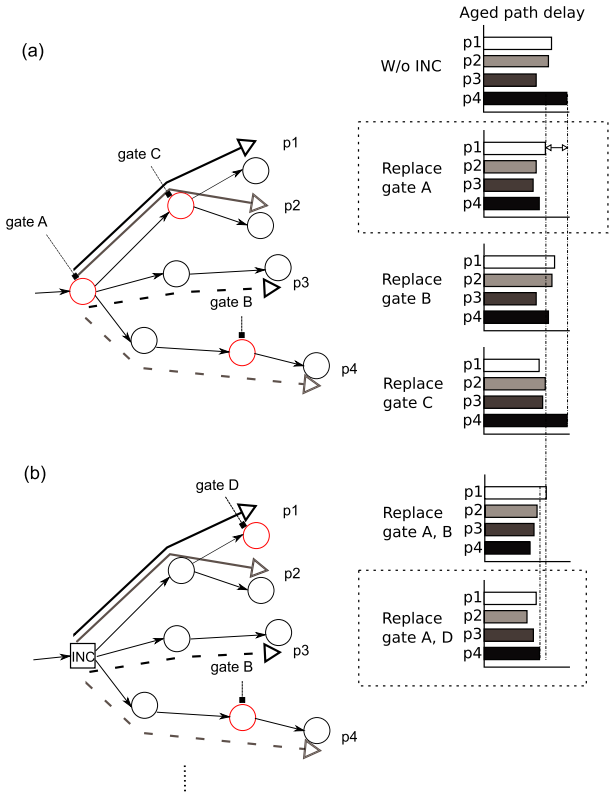

Fig. 4 Example of the INC logic replacement by COMP method.

NBTI mitigation than that of Fig. 3(a), mostly because of the probability updates and path delay evaluations.

In Fig. 3(b), M gates in total are serially replaced with INC logic from the N stress gates. In order to reflect the probability change by the replacement, signal probability is recomputed for all gates in the target circuit, and path delays are evaluated to find the best candidate. The replacement candidate is given to be the one that minimizes the worst delay after aging.

An example of the gate replacement in COMP method is shown in Fig. 4. This example contains four paths: p1, p2, p3, and p4, and three stress gates: gate A, gate B, and gate C. In the inner loop of Fig. 3(b), each stress gate is replaced with the INC logic and all the aged path delays including the INC logic gate are evaluated as shown in the bar charts on the left side of Fig. 4(a). Since all the aged path delays are changed by the INC logic replacement, the bar charts with different replacements also differ with each other. After the test replacement for each gate, the gate that achieves the best aged delay, gate A in this example, is chosen for the replacement. Once gate A is replaced, the signal probability is updated and the list of the stressed gates is also updated accordingly. This process is repeated for M times as shown in Fig. 4(b).

As can be seen, this process is very CPU intensive because both signal probability calculation and aged delay calculation for all paths have to be conducted in the innermost loop of the optimization process. In the next section, we propose a method to shorten these calculations.

4. Acceleration of INC Replacement by Path Clustering

In this section, the proposed method, through which we can efficiently find the replacement candidates, is explained. The outline of the proposed method is shown in Fig. 3(c). The key idea for achieving the acceleration is to limit the number of paths, and thus associated gates, which we evaluate for the mitigation effectiveness. For that purpose, we elaborated on 1) selecting *representative path* through path clustering, and 2) pursuing *gate pruning* to further skip probability and delay calculations.

Similar to the existing methods, we start from the DAG of the target circuit as shown in Fig. 3(c). Consequently, two-stage path clustering is conducted to balance clustering efficiency and accuracy. First, the paths are coarsely clustered ((A) in Fig. 3(c)) into several groups by using a fast graph-based clustering algorithm. Path signature is then calculated (B) to help form detailed path clusters (C). Here, the path signature that represents the similarity of the paths, i.e., sharing many common logic gate instances increases similarity, is introduced. From each cluster, a representative path is selected according to the aged path delay (D). The representative path is the one that becomes the most critical in each cluster after aging. Sequentially, the replacement gate is selected (E). We limit the evaluation of delay mitigation to only the stress gates that are on the representative paths or at the side-inputs of the critical paths. The gate that gives best mitigation result for the worst delay path will be selected as the replacement gate.

Because the representative paths are composed of the gate instances that are also the part of other paths in the same cluster, INC replacement along the representative path is expected to be also effective to mitigate the delay of other paths. Every time the candidate gate is selected from along the representative path, the new representative path is selected according to the aged timing. The updates of stress probability and that of representative path ensure effective mitigation in the worst timing of the circuit. Even when the change of worst delay path should occur during the optimization, the next INC replacement will shorten the delay of the newly became worst path.

In addition to introducing the representative path, we also utilize gate pruning. The evaluation of mitigation effect of a gate is skipped, when the replacement of the gate does not contribute to the most critical path delay at that time. The judgment can be realized by recording the dependency of paths to each gate replacement.

Hereafter, the five procedures, (A)–(E), in the proposed method will be described in detail.

(A) Coarse Path Clustering

In general, clustering requires evaluating the distances between all combinations of the elements. Hence, calculation time increases quadratically to the number of elements, and good path clustering becomes gradually difficult as element size becomes large. In order to make the proposed method

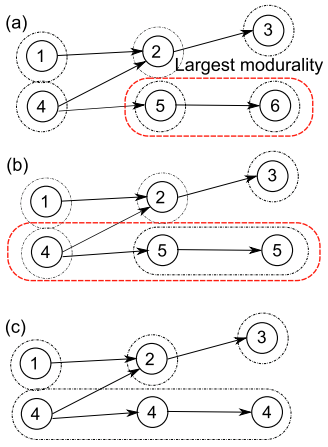

Fig. 5 Example of path clustering by Newman algorithm.

to be applicable to large circuits, we propose to split the path clustering process into two stages. In the first stage, Newman algorithm [7] is utilized. This is a fast graph-based clustering algorithm which constructs clusters according to the structure of the DAG. The clusters of the maximum modularity are merged. Here, the modularity between clusters i and j is,

$$\Delta Q = (e_{ij} + e_{ji} - 2a_i a_j), \tag{3}$$

where e_{ij} is the fraction of all edges from i to j, and $a_i = \sum_j e_{ij}$. By evaluating the modularity, the Newman algorithm suggests the number of clusters without requiring additional parameters for clustering.

An example of path clustering by Newman algorithm is shown in Fig. 5. Newman algorithm is one of the bottom-up clustering algorithms. The clustering algorithm starts by considering all the gates as individual clusters. First, the modularity ΔQ for all gates are calculated by Eq. (3). In Fig. 5(a), the modularity between the clusters '5' and '6' become the highest, where $e_{5,6} = e_{6,5} = 1/10$, $a_5 = 2/10$, and $a_6 = 1/10$. Then the two clusters are merged to be cluster '5' as shown in Fig. 5(b). This merging step is repeated until ΔQ becomes 0. In the next iteration, as shown in Fig. 5(c), the clusters '4' and '5' are merged in the same manner, where $e_{4,5} = e_{5,4} = 1/8$, $a_4 = 2/8$, $a_5 = 1/8$.

After all gates are labeled with their cluster numbers, the paths are clustered according to the labels of the gates in each path. An example of this clustering stage is shown in Fig. 6. In this example, the nodes represent gates and the arcs represent signal flow of the DAG. The gates in this example have been clustered into four, and the cluster number of each gate is shown in the node. At the end point of each path, the cluster numbers are enumerated in the order of the gates in the signal flow. According to the enumerated labels, path clusters are formed. In this example, paths p1 and p2 will be in the same cluster, and other two are in separate clusters.

(B) Path Signature Calculation

In order to subdivide clusters, we defined the path signature as a vector of gate instances. The element of the path signature is either 1 or 0. The path signature $\boldsymbol{x}_i$ for path i is

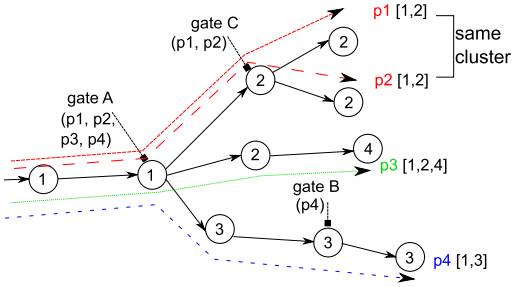

Fig. 6 Example of coarse path clustering and gate pruning.

$$\boldsymbol{x}_i = \Big(f_i(G_1), f_i(G_2), \cdots, f_i(G_j)\Big). \tag{4}$$

Here, $f_i(G_j)$ is 1 when the j-th gate is included in the i-th path, and 0 when the j-th gate is not included in that path.

(C) Cluster Refinement

The path clusters obtained in the first stage is further divided into smaller clusters in the second stage. We adopt an agglomerative hierarchical clustering [8], which is a hierarchical clustering algorithm utilizing a bottom-up approach. Starting from each element being a single cluster, the algorithm repeatedly merges the closest two clusters into a new cluster until the number of clusters reaches to k. Here, the extraction rate γ is the user-defined parameter to determine k as a product of γ and the number of target paths. We utilize Ward's method [11] to measure the distance between two clusters. When two clusters, u and v, are evaluated, the distance is defined as $d_{ss}(u \cup v) - d_{ss}(u) - d_{ss}(v)$, where $d_{ss}(u)$ is the sum of squares of the distances between the centroid and the elements of the cluster u. $d_{ss}(u \cup v)$ is the sum of squares of the distances between the centroid and each element.

(D) Representative Path Selection

The representative paths are extracted from each cluster based on aged path delay. Through the timing analysis considering NBTI degradation [12], a path that becomes the most critical in the path cluster after aging is selected to be the representative path. In the timing analysis, degraded path delay is computed by assuming threshold voltage degradation through Eqs. (1) and (2). As previously explained, the representative path is recalculated in every loop of the optimization process to reflect the change of the aged critical paths.

(E) INC Logic Replacements with Gate Pruning

The replacement candidate with the INC logic is selected among the stress gates. Through exhaustive trial, the gate that achieves the highest mitigation effect will be replaced. By repeating step (E) for M times, M logic gates are replaced with the INC logic. In determining the best replacement candidate, replacement efficacies of all the stress gates are evaluated through probability propagation and aged delay

calculation. These calculations are repeated for L' times (which is the number of stress gates) to determine a replacement gate, and thus occupy most of the optimization time.

To accelerate the most computationally intensive calculations conducted inside the innermost loop, *gate pruning* is devised. In general, a gate replacement does not necessarily mitigate timing of all paths. The affecting paths are entirely determined by the structure of DAG, and should be considered almost constant regardless of the circuit size. With this observation, the paths that are affected by the replacement of a gate can be stored in advance. By using the stored information, probability update and delay calculation can be skipped for the stress gate which does not affect the most critical path in that loop. Utilizing the gate pruning, only L gates are evaluated. Figure 6 also shows an example of gate pruning. The stress gates, gate A, B, and C, are associated with the affecting paths. When path p1 is timing critical, probability propagation and delay calculation for gate B can be safely skipped. If the critical path changes to p4 later, only gate B should be evaluated.

The aged path delay calculation, which is the most time consuming process, consists of two parts: the signal probability propagation T_{pp}, and aged path delay calculation with annotated signal probabilities T_{delay}. Without the representative path selection, all stress gates in the target circuit have to be evaluated. The computational time of this naive implementation becomes $T_{\mathrm{COMP}} = N(T_{\mathrm{pp}} + T_{\mathrm{delay}})$, where N is the number of the stress gates. In contrast, with the proposed method, the number of paths is reduced from n_{path} to ℓ, and the number of stress gates is significantly reduced from N to L. From this reduction, the computation time T_{PCCP} becomes $T_{\mathrm{PCCP}} = L\left(T_{\mathrm{pp}} + \frac{\ell}{n_{\mathrm{path}}} T_{\mathrm{delay}}\right) + T_{\mathrm{cluster}} + T_{\mathrm{prune}}$, here, T_{cluster} and T_{prune} are the time required for the computation of path clustering and the gate pruning, respectively.

5. Numerical Experiment

5.1 Experimental Setup

In order to quantitatively evaluate the effectiveness of the proposed method, numerical experiments are conducted using a five-stage pipelined RISC processor, which is generated by a commercial processor compiler [13], and s38584 from ISCAS'89 benchmark circuit [14]. In our evaluation, three methods shown in Fig. 3 are compared. The circuits are synthesized by a logic-synthesis tool [15] using Nangate 45 nm Open Cell Library [10]. The gate counts of the target circuits are summarized in Table 1. Here, the gate count is in equivalent two-input NAND gates, which is obtained by dividing the cell area of the target designs by the area of the smallest two-input NAND gate, 0.798 μm^2. The aged path delay is calculated by the NBTI-aware STA [12] under the condition of 400 K and 10 years of aging. The signal probabilities of the gates are calculated by a power analysis tool [16] and probability propagation is done by an in-house tool, respectively. The numbers of stressed gates of the RISC processor and s38584 are 1,075 and 1,575, respectively, when no INC logic replacement is applied. Paths are enumerated by an STA tool [17]. All experiments were carried out on Linux PC with Intel Xeon E5-2630 v2 2.60 GHz CPU, using a single thread. The proposed method was implemented using Python language.

Table 1 Gate counts of the target design in two-input NAND equivalents.

	RISC Processor	s38584
Combinational area	26,184	10,005
Sequential area	8,617	7,367
Total area	34,800	17,372

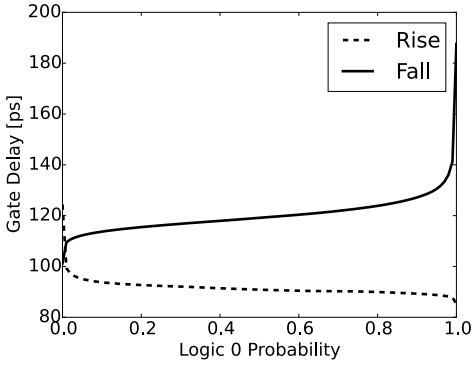

Fig. 7 Relationship between the signal probability and the rise and fall input delay of a two-input AND gate.

The probability calculations are conducted in two steps; first at the gate level, then at the transistor level. The transistor level analysis is required because NBTI of the multi-stage gates affects both rise and fall delays due to the inverting nature of CMOS gates, while that of the single-stage gates affects only to the fall delay. Figure 7 shows the delay of the smallest two-input AND gate, which is a two-stage gate, as a function of input signal probability while that of another input pin is 0.5. Under this effect, extremely low or high signal probabilities are concerning. However, in this work, only the high side of the signal probability is considered for mitigation because 1) it halves the number of gates that needs to be considered, 2) delay degradation of the fall input is dominant for most of the logic gates, and 3) sufficient mitigation can be obtained owing to probability propagation. One can possibly include both sides to obtain better optimization results than this work.

Before starting the experiments, based on the bimodal distribution of fresh path delay, the paths having faster delay were filtered out. The number of the remaining paths is 25,446, which will be a common input for all the methods.

5.2 Delay Mitigations

Tables 2 and 3 show the optimization results of the GA-based method (GA), comprehensive method (COMP), and the proposed method (PCCP; path clustering and candidate pruning) for the RISC processor and s38584, respectively. Worst-path delays after 10 years of aging are presented. If no mitigation is applied, the worst-path delays become 7.14 ns

Table 2 Comparison of the mitigated worst path delay (RISC processor).

Method	Mitigated worst path delay [ns]
GA [6]	5.78 (M = 200)
Comprehensive (COMP)	5.15 (M = 50)
Proposed (PCCP)	5.17 (M = 50)
No mitigation	7.14

Table 3 Comparison of the mitigated worst path delay (s38584).

Method	Mitigated worst path delay [ns]
GA [6]	1.71 (M = 200)
Comprehensive (COMP)	1.73 (M = 25)
Proposed (PCCP)	1.74 (M = 25)
No mitigation	2.22

and 2.22 ns in the RISC processor and s38584, respectively, after the same period of aging. By applying the mitigation methods, delay increase due to aging has been substantially mitigated. In particular, COMP and PCCP give better or similar mitigation results than GA with a small number of INC logic replacements. We obtain two important implications from the result of the RISC processor: 1) updating the signal probability and path delay during optimization is critically important, and 2) there may be much room for further optimizing INC logic replacement. Note that, in the case of s38584, the mitigation result by GA is slightly better than those of COMP and PCCP. This is because an INC logic replacement does not change the signal probability of other part of the circuit compared to the RISC processor. Hence, the effectiveness of the probability update is not significant in s38584.

COMP and PCCP yield similarly better mitigation results than GA, showing that the proposed path clustering is effective in the RISC processor, and the important gates that should be replaced with the INC logic are successfully preserved. In the following, close comparisons are made on the results of COMP and PCCP.

Figure 8 shows the mitigated worst path delay of the RISC processor as a function of the number of replacements M by COMP and PCCP with $\gamma = 0.002$ and 0.005. It is shown that the worst path delay decreases as M increases, and there is almost no difference between the two methods. This indicates that almost the same mitigation gains can be obtained from both methods.

5.3 Acceleration through Path Grouping

Figures 9 and 10 show the ratio of optimization time as a function of the extraction rate γ for the RISC processor and s38584, respectively. The number of INC logic replacements M is 50 in the RISC processor and 25 in the s38584 circuit, respectively. The left-axis shows the ratio of computational time between COMP and PCCP. As the extraction rate in PCCP increases, the number of representative paths increases, and then the number of gates to be evaluated also increases, leading to longer optimization time.

Shown on the right axis is the *replacement accuracy* of PCCP as compared to COMP. Here, the accuracy

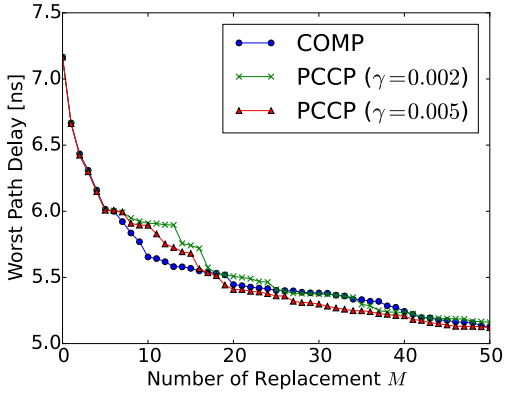

Fig. 8 Relationship between the number of replacements M and the mitigated worst path delay of COMP and PCCP.

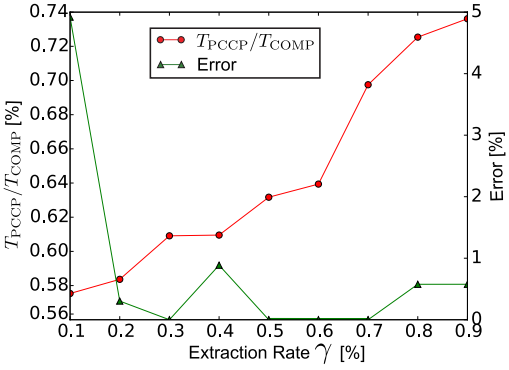

Fig. 9 Ratio of computational time and timing mitigation accuracy as functions of the extraction rate γ in the RISC processor, at replacement count M = 50. Ratio is taken between the PCCP and COMP. Error is defined as being the result of COMP as reference.

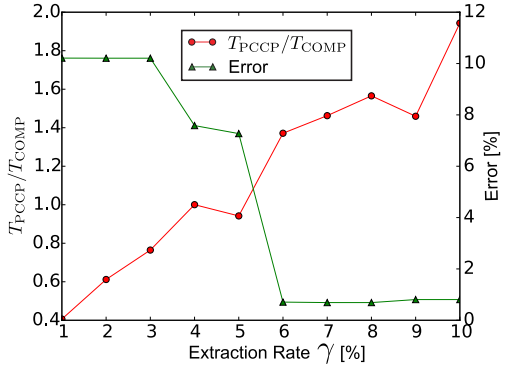

Fig. 10 Ratio of computational time and timing mitigation accuracy as functions of the extraction rate γ in s38584, at a replacement count M = 25. Ratio is taken between the PCCP and COMP. Error is defined as being the result of COMP as reference.

is represented by the error defined as error = $(D_{\mathrm{PCCP}} - D_{\mathrm{COMP}})/(D_{\mathrm{COMP}})$, where D_{PCCP} and D_{COMP} are the aged worst path delays of the PCCP and COMP, respectively. When the extraction rate is small, a larger number of paths are contained in small number of clusters, and the one representative path of a cluster does not be a good representative, since the gates that are not on the representative paths are not

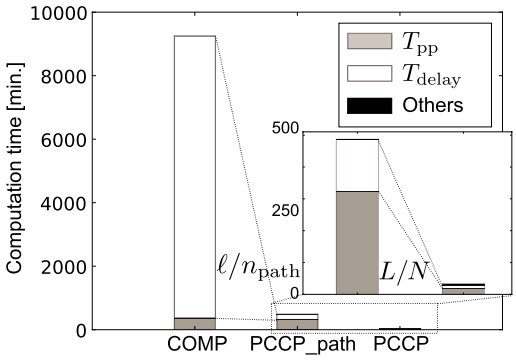

Fig. 11 Breakdown of the computation time of COMP and PCCP.

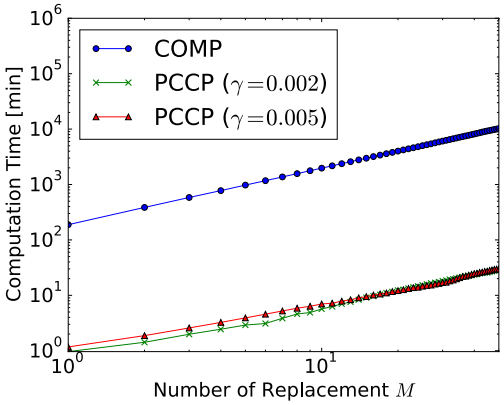

Fig. 12 CPU time of COMP and PCCP as a function of the number of gate replacement count M.

considered for INC replacement, the chance of selecting non-optimal gates increases. However, regardless of the choice of γ, the proposed method yields very close result with COMP. Considering the small error between PCCP and COMP, the representative paths successfully limit the number of candidate gates, while preserving the effective gates for INC logic replacement. The mitigation errors for the RISC processor and s38584 are smaller than 1% at $\gamma = 0.002$ and 0.06, respectively. The numbers of extracted paths are 233 and 1,020 for the RISC processor and s38584, respectively. In the case of RISC processor, we can reduce the calculation time to 0.584%, which is 171 times speed up while maintaining 0.304% error. The computation time is reduced from 9,246 to 54 minutes. The computation time for s38584 is also reduced from 2,455 to 34 minutes within 0.708% error.

Figure 11 shows the breakdown of the calculation time for the RISC processor at $\gamma = 0.002$. Cumulative times for both probability propagation (T_{pp}) and path delay calculation (T_{delay}) are shown. In COMP, T_{delay} is a dominant portion of the total calculation time. It has been reduced from 8,877 to 163.5 minutes by limiting the number of target paths from n_{path} to ℓ (25,446 to 233, in this example) through the representative path extraction. This reduction is presented in the middle as PCCP_path. T_{pp} does not change by the clustering because the probability propagation has to be conducted over the entire circuit. As shown in the right, ($T_{delay} + T_{pp}$) is further reduced because the number of evaluated stress gates is reduced from N to L by limiting the paths. In this example, the average values of N and L are 1,056 and 61, respectively. In PCCP, time required for path clustering ($T_{cluster}$) and candidate pruning characterization (T_{prune}) is newly required, but they are 33 seconds and 294 seconds, which can be considered ignorable.

Figure 12 shows the computation time for the RISC processor as the function of M under the same condition with Fig. 8. The computation time is almost linearly in M, and PCCP is much faster than COMP as shown in Fig. 11. The results in Figs. 8 and 12 shows that the proposed method can accelerate the computation while maintaining the mitigation gain equivalent to COMP.

6. Conclusion

In this paper, we proposed a path clustering approach for accelerating logic gate replacements with INC logic. In the proposed technique, through the Newman clustering and the agglomerative hierarchical clustering, representative paths are selected from each group to limit the number of gates to consider the replacement and that of paths to evaluate the aged path delay. Also, by pruning gates which are not affected in signal probability of aged critical paths by the INC logic replacement, calculation times of the signal probability and the aged path delay are saved. The experimental evaluation using a RISC processor demonstrates that the proposed technique achieves 171 times speedup for selecting INC replacement candidates while maintaining the mitigation error within 0.304% compared with the case without the representative path extraction.

Acknowledgements

This work was partially supported by JSPS KAKENHI Grant No. 26280014 and 15K15960. The authors also acknowledge support from VLSI Design and Education Center (VDEC), the University of Tokyo in collaboration with Synopsys, Inc.

References

[1] W. Wang, S. Yang, S. Bhardwaj, S. Vrudhula, F. Liu, and Y. Cao, "The impact of NBTI effect on combinational circuit: Modeling, simulation, and analysis," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.18, no.2, pp.173–183, 2010.
[2] S. Bhardwaj, W. Wang, R. Vattikonda, Y. Cao, and S. Vrudhula, "Predictive modeling of the NBTI effect for reliable design," Proc. IEEE Custom Integrated Circuits Conference, pp.189–192, 2006.
[3] H. Konoura, Y. Mitsuyama, M. Hashimoto, and T. Onoye, "Comparative study on delay degrading estimation due to NBTI with circuit/instance/transistor-level stress probability consideration," Proc. IEEE International Symposium on Quality Electronic Design, pp.646–651, 2010.
[4] D.R. Bild, R.P. Dick, and G.E. Bok, "Static NBTI reduction using internal node control," ACM Trans. Des. Autom. Electron. Syst.,

vol.17, no.4, pp.45–75, 2012.
[5] Y. Wang, X. Chen, W. Wang, Y. Cao, Y. Xie, and H. Yang, "Leakage power and circuit aging cooptimization by gate replacement techniques," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.19, no.4, pp.615–628, 2011.
[6] S. Bian, M. Shintani, Z. Wang, M. Hiromoto, A. Chattopadhyay, and T. Sato, "Runtime NBTI mitigation for processor lifespan extension via selective node control," Proc. IEEE Asian Test Symposium, pp.234–239, 2016.
[7] M.E.J. Newman, "Fast algorithm for detecting community structure in networks," Phys. Rev. E, vol.69, no.6, p.066133, 2004.
[8] M.R. Anderberg, Cluster Analysis for Applications: Probability and Mathematical Statistics: A Series of Monographs and Textbooks, Academic Press, 1973.
[9] S. Morita, S. Bian, M. Shintani, M. Hiromoto, and T. Sato, "Path grouping approach for efficient candidate selection of replacing NBTI mitigation logic," Proc. 20th Workshop on Synthesis And System Integration of Mixed Information Technologies, pp.225–230, 2016.
[10] Si2.org, "Nangate 45 nm open cell library," Available: http://www.si2.org
[11] J.H. Ward, Jr., "Hierarchical grouping to optimize an objective function," J. Am. Stat. Assoc., vol.58, no.301, pp.236–244, 1963.
[12] S. Bian, M. Shintani, S. Morita, M. Hiromoto, and T. Sato, "Nonlinear delay-table approach for full-chip NBTI degradation prediction," Proc. IEEE International Symposium on Quality Electronic Design, pp.307–312, 2016.
[13] Synopsys, Inc., Processor Designer G-2012.09.
[14] F. Brglez, D. Bryan, and K. Koiminski, "Combinational profiles of sequential benchmark circuits," Proc. IEEE International Symposium on Circuits and Systems, pp.1929–1934, 1989.
[15] Synopsys, Inc., Design Compiler I-2013.06.
[16] Synopsys, Inc., PrimeTime PX H-2013.06.
[17] Synopsys, Inc., PrimeTime Fundamental H-2013.06.

Shumpei Morita received B.E. degree in Electrical and Electronic Engineering from Kyoto University in 2016. He is a master course student at Department of Communications and Computer Engineering, Kyoto University.

Song Bian received B.S. from University of Wisconsin-Madison, Wisconsin, USA in 2014. He attended Kyoto University in 2015 in pursuing a master's degree in the field of computer engineering. His main areas of interest include electric design automation (EDA), processor architecture and semiconductor reliability.

Michihiro Shintani received B.E. and M.E. degrees from Hiroshima City University, Hiroshima, Japan, and a Ph.D. degree from Kyoto University, Kyoto, Japan, in 2003, 2005, and 2014, respectively. He was with the Panasonic Corporation, Osaka, Japan, from 2005 to 2014, and at the Semiconductor Technology Academic Research Center (STARC), Yokohama, Japan, from 2008 to 2010. In 2014, he joined the Graduate School of Informatics, Kyoto University, where he is currently a program-specific researcher. His research interests include reliability-aware LSI design and simulation of power electronics circuit. He is currently a member of IEEE and Institute of Electronics, Information and Communication Engineers (IEICE). He received the Best Paper Award at the IEEE Workshop on RTL and High Level Testing (WRTLT) 2004.

Masayuki Hiromoto received B.E. degree in Electrical and Electronic Engineering and M.Sc. and Ph.D. degrees in Communications and Computer Engineering from Kyoto University in 2006, 2007, and 2009 respectively. He was a JSPS research fellow from 2009 to 2010, and with Panasonic Corp. from 2010 to 2013. In 2013, he joined the Graduate School of Informatics, Kyoto University, where he is currently an assistant professor. His research interests include VLSI design methodology, image processing and pattern recognition. He is a member of IEEE and IPSJ.

Takashi Sato received B.E. and M.E. degrees from Waseda University, Tokyo, Japan, and a Ph.D. degree from Kyoto University, Kyoto, Japan. He was with Hitachi, Ltd., Tokyo, Japan, from 1991 to 2003, with Renesas Technology Corp., Tokyo, Japan, from 2003 to 2006, and with the Tokyo Institute of Technology, Yokohama, Japan. In 2009, he joined the Graduate School of Informatics, Kyoto University, Kyoto, Japan, where he is currently a professor. He was a visiting industrial fellow at the University of California, Berkeley, from 1998 to 1999. His research interests include CAD for nanometer-scale LSI design, fabrication-aware design methodology, and performance optimization for variation tolerance. Dr. Sato is a member of the IEEE and the Institute of Electronics, Information and Communication Engineers (IEICE). He received the Beatrice Winner Award at ISSCC 2000 and the Best Paper Award at ISQED 2003.